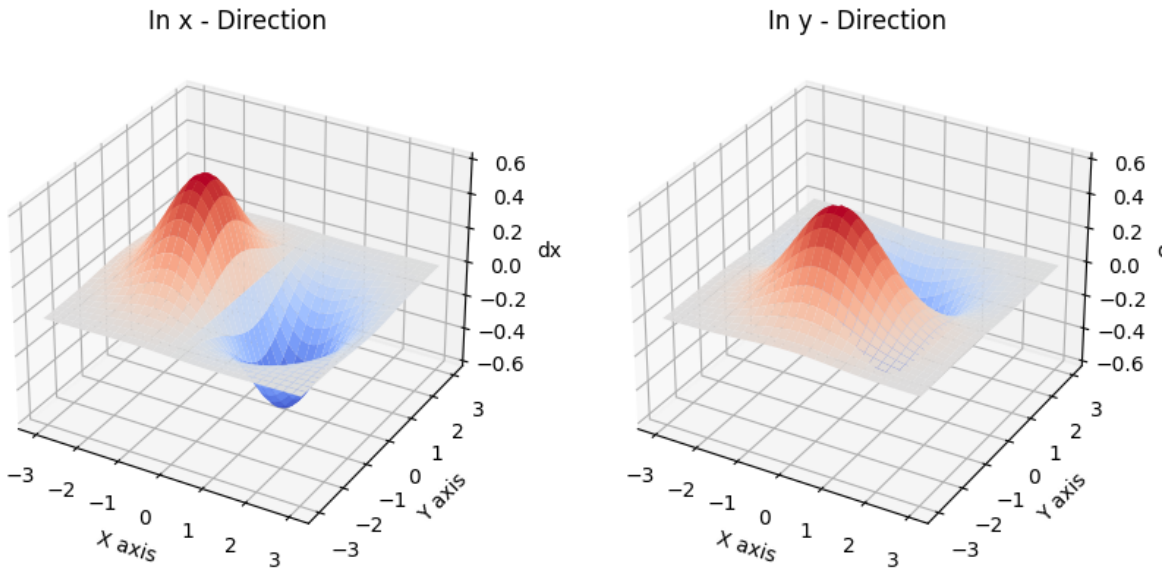
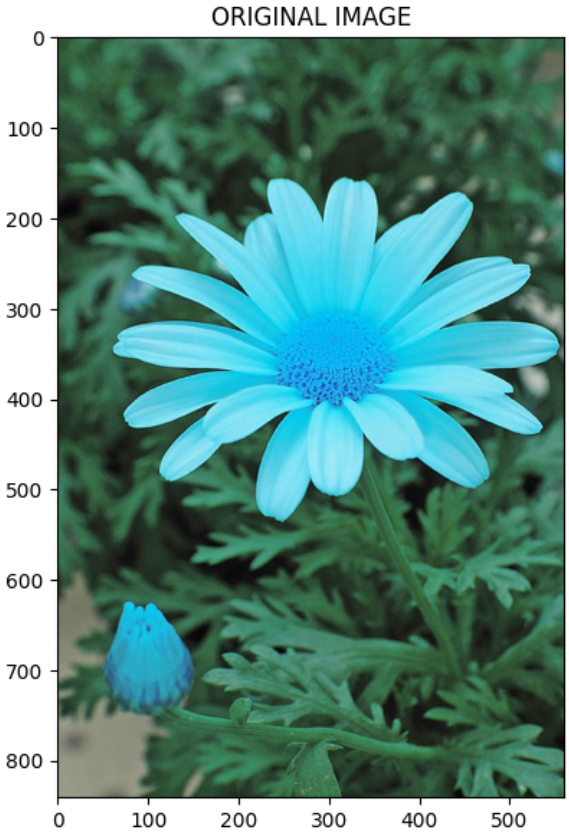
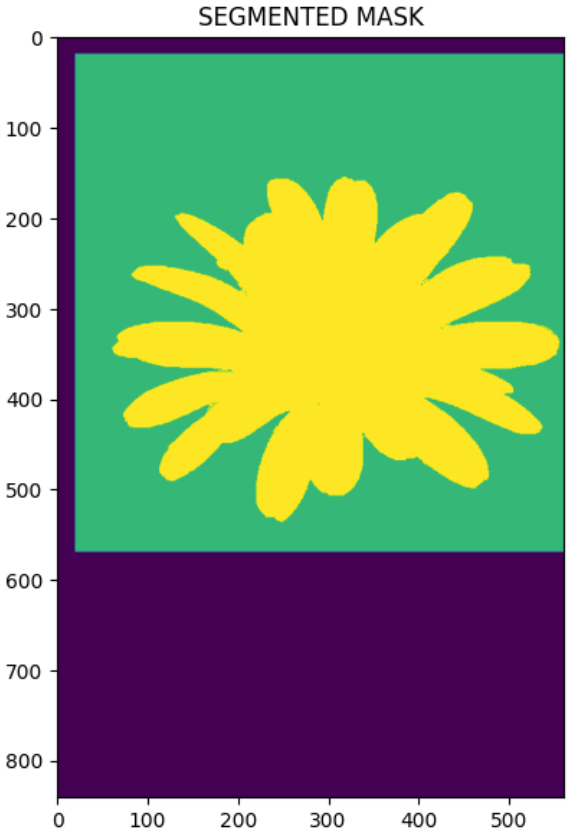
In [ ]:
```
#Question 02
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

x = y = np.arange(-3, 3.1, 0.1)
X, Y = np.meshgrid(x, y)
sigma = 1
# Derivative of gaussian kernels WRT to X and Y
dx = (-X / sigma ** 2) * np.exp(-(X ** 2 + Y ** 2) / (2 * sigma ** 2))
dy = (-Y / sigma ** 2) * np.exp(-(X ** 2 + Y ** 2) / (2 * sigma ** 2))
fig = plt.figure(figsize=(10, 5))
ax = fig.add_subplot(1, 2, 1, projection='3d')
ax.plot_surface(X, Y, dx, cmap='coolwarm')
ax.set_title('In x - Direction')
ax.set_xlabel('X axis')
ax.set_ylabel('Y axis')
ax.set_zlabel('dx')
ax = fig.add_subplot(1, 2, 2, projection='3d')
ax.plot_surface(X, Y, dy, cmap='coolwarm')
ax.set_title('In y - Direction')
ax.set_xlabel('X axis')
ax.set_ylabel('Y axis')
ax.set_zlabel('dy')

plt.show()
```

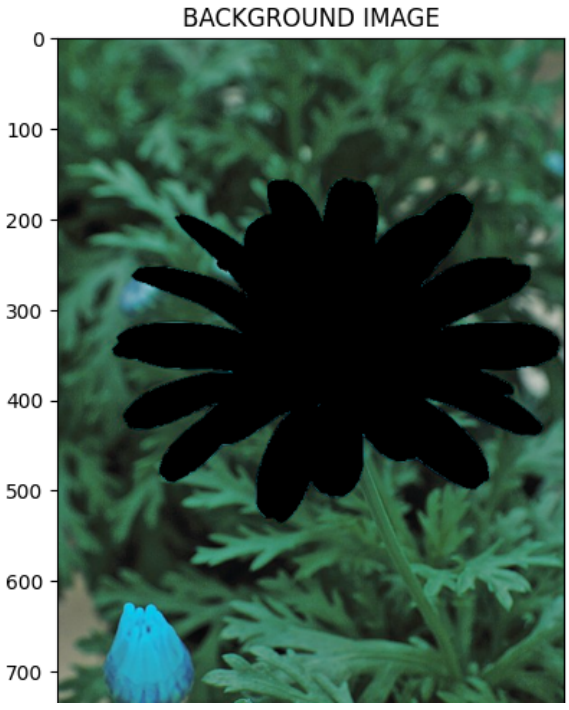


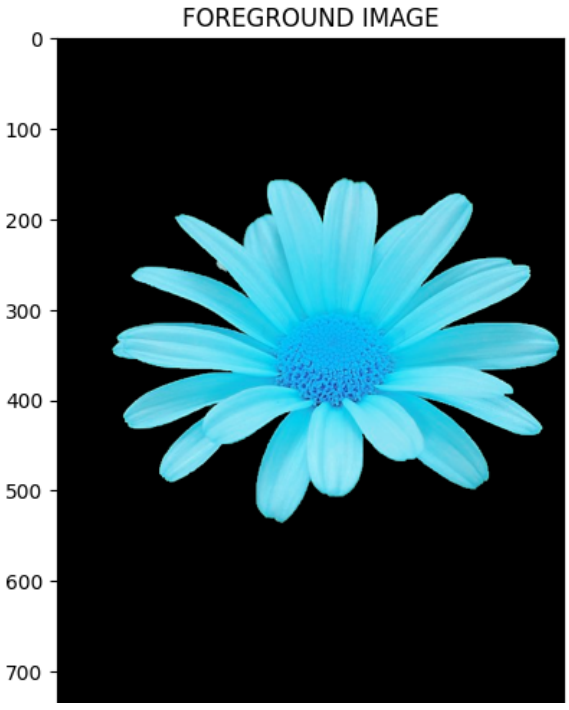
In [ ]:
```
# Question 04
import cv2 as cv
import matplotlib.pyplot as plt
import numpy as np
im = cv.imread('/examples/daisy.jpg')
assert im is not None
rectangle = (20,20,550,550)
mask = np.zeros(im.shape[:2], np.uint8)
background = np.zeros((1,65), np.float64)
```

```
cv.grabCut(im, mask, rectangle, background, None, 5, cv.GC_INIT_WITH_RECT)
newmask = np.where((mask == cv.GC_FGD) | (mask == cv.GC_PR_FGD), 1, 0).astype('uint
foreground = cv.bitwise_and(im, im, mask=newmask)
background = cv.bitwise_and(im, im, mask=1 - newmask)
blurred_background = cv.GaussianBlur(background,(31,31),0)
enhanced_image = cv.addWeighted(foreground, 1, blurred_background, 0.8, 0)
fig ,ax = plt.subplots (2,2, figsize= (10,20))
ax[0,0].imshow(im,cmap = 'gray')
ax[0,0].set_title('ORIGINAL IMAGE')
ax[0,1].imshow(mask)
ax[0,1].set_title('SEGMENTED MASK')
ax[1,1].imshow(foreground)
ax[1,1].set_title('FOREGROUND IMAGE')
ax[1,0].imshow(background)
ax[1,0].set_title('BACKGROUND IMAGE')
plt.figure(figsize=(12,6))
plt.subplot(1, 2, 1), plt.imshow(im[:,:,::-1]), plt.title('ORIGINAL IMAGE')
plt.axis('off')
plt.subplot(1, 2, 2), plt.imshow(enhanced_image[:,:,::-1]), plt.title('ENHANCED IMA
plt.axis('off')
plt.show()
```


ORIGINAL IMAGE
0
100
200
300
400
500
600
700
800
0
100
200
300
400
500


SEGMENTED MASK
0
100
200
300
400
500
600
700
800
0
100
200
300
400
500


BACKGROUND IMAGE
0
100
200
300
400
500
600
700

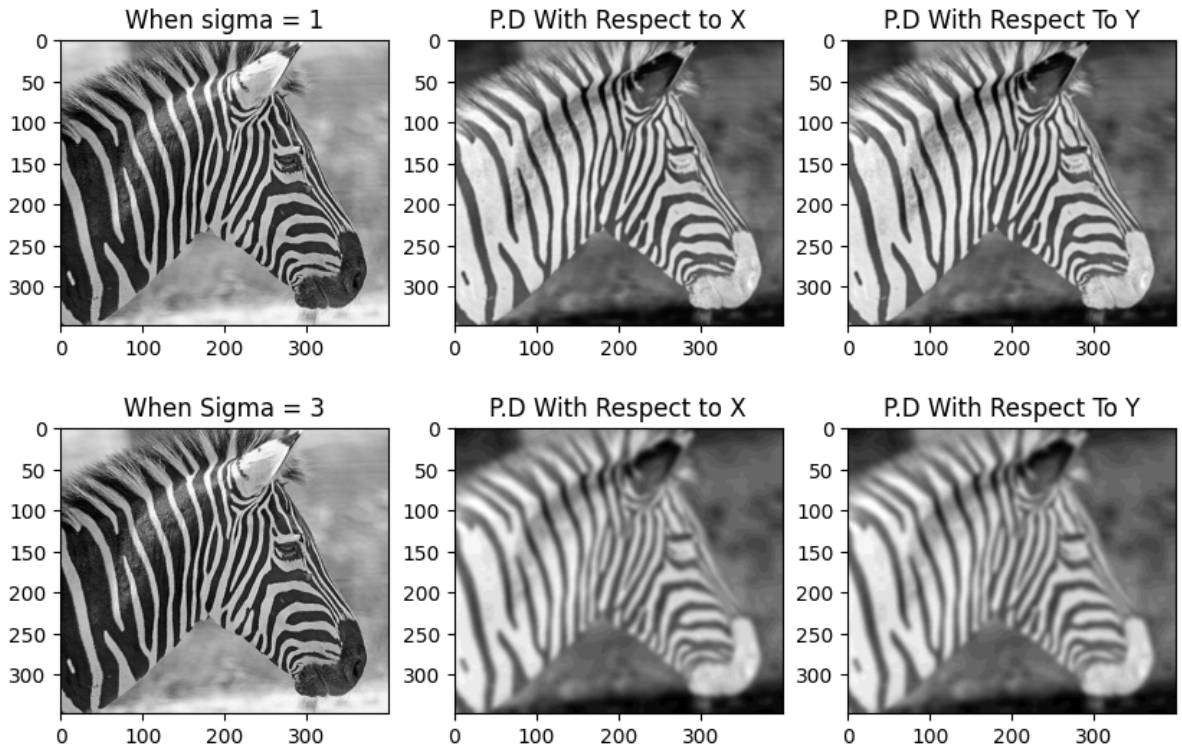

FOREGROUND IMAGE
0
100
200
300
400
500
600
700

ORIGINAL IMAGE

ENHANCED IMAGE

In [ ]:
```
#Question 03
import numpy as np
import cv2 as cv
from scipy import ndimage
import matplotlib.pyplot as plt
im = cv.imread('/examples/zebrahead.jpg', cv.IMREAD_GRAYSCALE)
sigma = 1
kernel = np.zeros((5*sigma, 5*sigma))
for i in range(5*sigma):
    for j in range(5*sigma):
        x = i - 2*sigma
        y = j - 2*sigma
        kernel[i, j] = 1/(2*np.pi*sigma**2) * np.exp(-(x**2+y**2)/(2*sigma**2))
dKdx = ndimage.convolve(im, -kernel/sigma**2)
dKdy = ndimage.convolve(im, -kernel.T/sigma**2)
fig, ax = plt.subplots(1, 3, figsize=(10, 5))
ax[0].imshow(im, cmap='gray')
ax[0].set_title('When sigma = 1')
ax[1].imshow(dKdx, cmap='gray')
ax[1].set_title('P.D With Respect to X')
ax[2].imshow(dKdy, cmap='gray')
ax[2].set_title('P.D With Respect To Y')
plt.show()
sigma = 3
kernel = np.zeros((5*sigma, 5*sigma))
for i in range(5*sigma):
    for j in range(5*sigma):
        x = i - 2*sigma
        y = j - 2*sigma
```

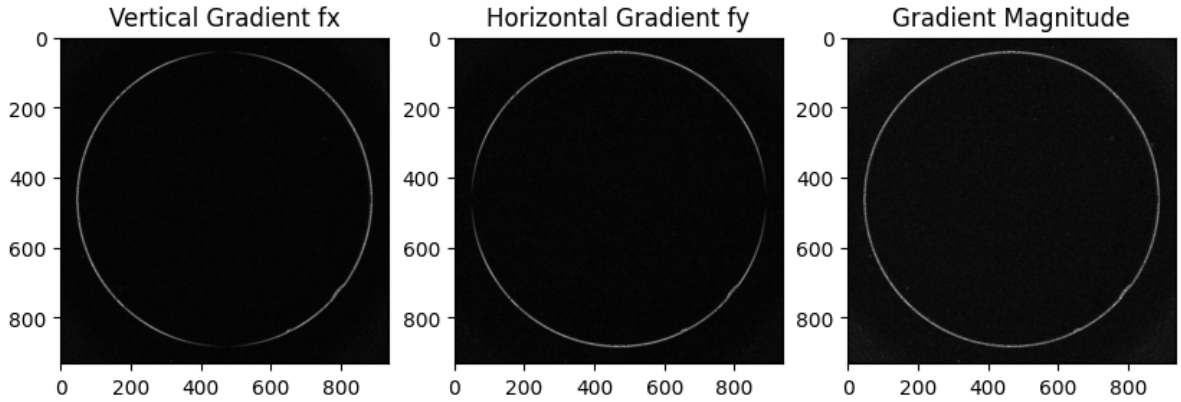
```
        kernel[i, j] = 1/(2*np.pi*sigma**2) * np.exp(-(x**2+y**2)/(2*sigma**2))
dKdx = ndimage.convolve(im, -kernel/sigma**2)
dKdy = ndimage.convolve(im, -kernel.T/sigma**2)
fig, ax = plt.subplots(1, 3, figsize=(10, 5))
ax[0].imshow(im, cmap='gray')
ax[0].set_title('When Sigma = 3')
ax[1].imshow(dKdx, cmap='gray')
ax[1].set_title('P.D With Respect to X')
ax[2].imshow(dKdy, cmap='gray')
ax[2].set_title('P.D With Respect To Y')
plt.show()
```



In [ ]:

```
# Question 1
import cv2 as cv
import numpy as np
import matplotlib.pyplot as plt
im = cv.imread('/examples/contact_lens.tif')
im_gray = cv.cvtColor(im, cv.COLOR_BGR2GRAY)
sobel_x = cv.Sobel(im_gray, cv.CV_64F, 1, 0)
sobel_y = cv.Sobel(im_gray, cv.CV_64F, 0, 1)
magnitude = cv.addWeighted(np.absolute(sobel_x), 1, np.absolute(sobel_y), 1, 0)
fig, ax = plt.subplots(1, 3, figsize=(10, 5))
ax[0].imshow(np.absolute(sobel_x).astype('uint8'), cmap='gray')
ax[0].set_title("Vertical Gradient fx")
ax[1].imshow(np.absolute(sobel_y).astype('uint8'), cmap='gray')
ax[1].set_title("Horizontal Gradient fy")
ax[2].imshow(magnitude.astype('uint8'), cmap='gray')
ax[2].set_title("Gradient Magnitude")
plt.show()
blurring = cv.GaussianBlur(im_gray, (5, 5), 0)
edges = cv.Canny(blurring, 50, 150)
y_cord, x_cord = np.where(edges != 0)
Center_X = int(np.mean(x_cord))
```

```
Center_Y = int(np.mean(y_cord))
print("Center :", "(",Center_X, "," ,Center_Y,")")
distances = np.sqrt((x_cord - Center_X)**2 + (y_cord - Center_Y)**2)
diameter = int(2 * np.max(distances))
cv.circle(im, (Center_X, Center_Y), 5, (0, 0, 255), -1)
cv.circle(im, (Center_X, Center_Y), diameter // 2, (0, 255, 0), 2)
cv.putText(im, f"Diameter is : {diameter}", (10, 30), cv.FONT_ITALIC, 0.6, (0, 0, 2
plt.imshow(cv.cvtColor(im, cv.COLOR_BGR2RGB))
plt.show()
```




```
Center : ( 467 , 463 )
```